# A 3D bus interconnect for network line cards

Jacob Engel and Taskin Kocak
Department of Electrical and Computer Engineering
University of Central Florida, Orlando, FL. 32816
e-mail: {jengel, tkocak}@cs.ucf.edu

*Abstract*—**In this paper, we propose a 3D bus architecture as a processor-memory interconnection system to increase the throughput of the memory system currently used on line cards. The 3D bus architecture allows multiple processing elements on a line card to access a shared memory. The main advantage of the proposed architecture is to increase the network processor off-chip memory bandwidth while diminishing the latency otherwise caused by the single bus competition.**

## I. Introduction

As network line rates are constantly increasing, memory access times keep decreasing. For example, a 40 byte packet arrive every 32 ns. Current network processors or network processing units (NPUs) use multithreading to hide memory latency. However, it is not clear whether this technique will scale well at 1 Tbps and beyond. In this paper, we report our initial work on processor-memory interconnections to increase the throughput of the memory system currently used on line cards. This can be easily done by increasing the memory bandwidth, however the current chip manufacturing techniques limit the number of I/O pins, and only so many of them can be memory I/Os.

Our interconnect bus structure is categorized as mezzanine interconnect which generally employs an address/data read/write data model with memory-like semantics and is targeted for simple translation between processor bus memory operations and mezzanine interconnects transactions [1]. Examples of mezzanine type interconnect architectures are: SPI-4.2 [3] which is a point-to-point communication architecture between MACs and NPUs or switch fabrics; CSIX [4], a common switch interface layer between NPU and switch fabric, and HyperTransport [5] which is a point-to-point, chip-to-chip interconnect technology that uses packet-based protocol and variable link width.

The main advantage of the proposed architecture is to increase the NPU off-chip memory bandwidth while diminishing the latency otherwise caused by the bus competition. Compared to parallel bus architectures, our 3D bus can carry more data on the wire at the same time. This is due to the fact that the bus from processor to memory is broken into shorter wires or links (see Fig. 1). Each link carries a different data set or memory packet. Furthermore, the 3D bus architecture can handle network traffic bursts by load-balancing write operations to different memory banks.

## II. The proposed architecture

Our proposed 3D interconnection architecture is shown in Fig. 2. The 3D bus structure is a packet-based multiple path forwarding mechanism that allows network packets to be shared by different processor and memory modules on the network line card. In Fig. 2, the line card processing and communication components which have access or require access to the memory banks are shown on the left. The components in the figure are given as example, and other functional components can be also interfaced to this bus. The memory banks are located on the other side of the 3D bus structure. Each component, which requires memory access, sends its data encapsulated in packets.

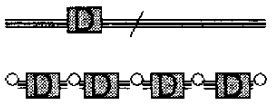

Fig. 1. Channels used in parallel and 3D bus

The default route is in the $x$-axis direction (shortest path). If there is a congested area (hot spot), packets in transit will take a different route in $y$-axis or $z$-axis directions using the traffic controller (TC) in each corner (node). The proposed architecture protocol utilizes an efficient message-passing structure to transfer data. If a link goes down, not only should the fault be limited to the link, the additional links from the intermediate nodes should ensure the connectivity continues.

### A. Routing Mechanism

The packets are routed to the destination memory module by following the header using wormhole routing. The message required to be sent to or from memory module is segmented into smaller size packets (flits). First the packet's header is sent to set the direction on each node in its path. The rest of the packets comprising the message do not wait, but transmitted in a pipeline manner following the message header as illustrated in Fig. 3. The packets size is determined by the channel width.

The main advantage in using wormhole routing in this 3D bus structure is that it diminishes the latency as the size of the message increases while increasing its throughput. The major part of the latency is hidden in the transfer of the first packet. The rest of the packets are following it and introducing only wire transfer delay. As the messages size increases, the ratio of consecutive latencies decreases. From a throughput viewpoint, the packets can enter the 3D bus structure from all four inputs with every $T$ sec following their header. $T$ is the propagation delay of one bit in one unit length which is equal to 62.5 ps per 1 cm using the current manufacturing

0-7803-8322-2/04/$20.00 ©2004 IEEE.

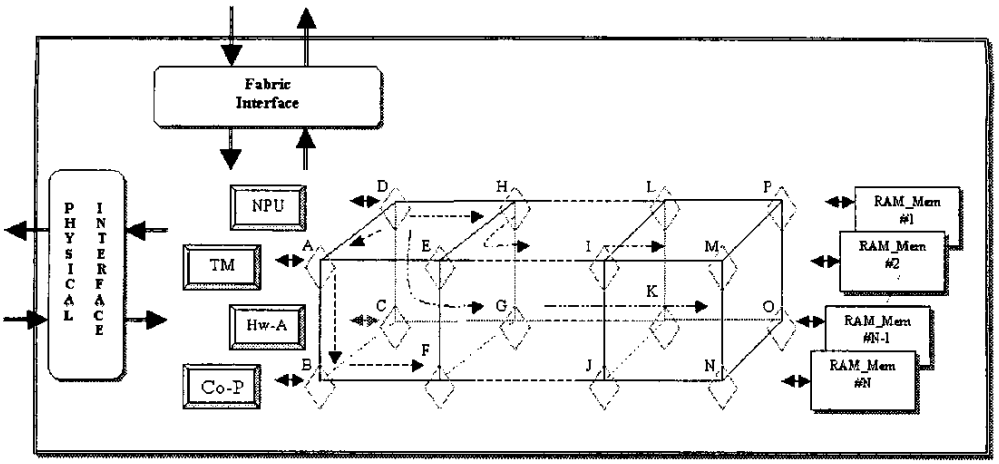


Fig. 2. 3D bus structure on the network line card

technology. This becomes a great advantage in achieving high throughput while a parallel bus can only send those packets like a store and forward type architecture (see Fig. 6).

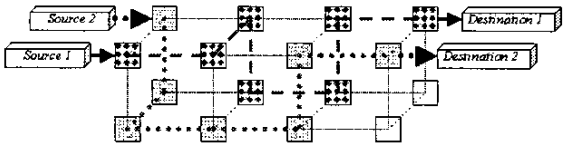


Fig. 3. Wormhole routing

## III. Performance Analysis

### A. Cube Notation

Cubes can be connected in series in order to increase the bus routing paths. Each vertical cube face is marked $i$ since it is incremented in the $x$-axis direction. Within each $i$ plane there are four corners, called $j$ notation, moving in a clockwise direction.

The $j$ notation has a dual digit value. Its first digit is the $i$ plane to which $j$ belongs and the other digit is the $j$ location within the plane. The $k$ notation is used to distinguish the horizontal buses connecting vertical $i$ planes. $k$ gets a four digit number. First two digits are the $i$ and $j$ on the left edge (source node) and the other two digits are the $i$ and $j$ values to the right of the bus (destination node).

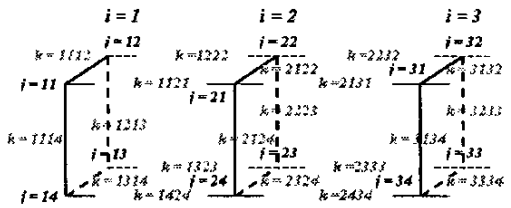


Fig. 4. 3D cube interconnection notation

Each node is located as the origin of axis in a three dimensional coordinates system. The movement of data words is translated to movement along the axis. A word moving from a bottom node up is moving in the positive $z$-direction and down in the negative $z$-direction. A packet moving from top node up, to its adjacent node in parallel, is moving along the positive $y$-direction. $X$ is the movement to the right (toward the output) or left (toward the input). A node transfers data word based of a default order of departure directions. First, it will try to send it in the positive $x$-direction. If the node is busy, then it will send it in the positive $y$-direction (up) and is this node is busy too it send it in the negative/positive $z$-direction (down for top node and up for bottom node). In case all adjacent nodes are busy the data word will not be sent until one of the routes becomes available.

### B. Shortest Path

The shortest path and longest path from a certain input node to destination was calculated by assigning labels on each node and bus segment as explained earlier (Fig. 4). The result, as shown in figure 5, was a hierarchical tree. Figure 5 represents only a single cube.

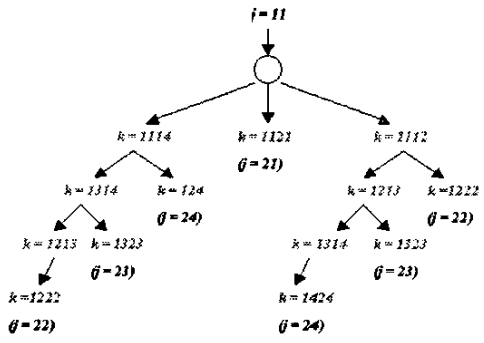


Fig. 5. Hierarchical Tree

We calculated the shortest path for multiple cubes (up to 4) by constructing incremental view of trees by taking the Manhattan distance from each corner. Each cube arch corresponded to new child node on the tree. We draw the complete tree for up to three cubes interconnection. The results we got show that the shortest path from input to output can be reached in three hops whereas, the longest path can be reached with up to twelve hops. Moreover, there is a linear relationship between the two by a factor of four. These results assist in performance evaluations of the best vs. worst latency and throughput results.

### C. Measures

We use standard performance metrics such as latency and throughput for evaluation. Latency is defined as the time it takes for a complete message to reach its destination. We adopt Dally's basic equation for $k$-ary $n$-cube interconnects [2] and modified it on our design. The resulting latency ($L$) equation is

$$L_{3D} = (\frac{M}{w} - 1 + D) * T + D * T(n) \quad (1)$$

where $M$ is the message size, $w$ is the channel width, $D$ is the Manhattan distance, $T$ represents the propagation delay of one bit in one unit length which is equal to 62.5ps (per 1 cm), and $T(n)$ is the node processing (switching) time. For the best case, $D = c$ and for the worst case, $D = 4 * c$. The header latency is larger than the rest of the message $(M/w-1)$ since it sets the nodes direction in time $T(n)$. The rest of the message just requires to propagate through the channels in $T$ time per unit length.

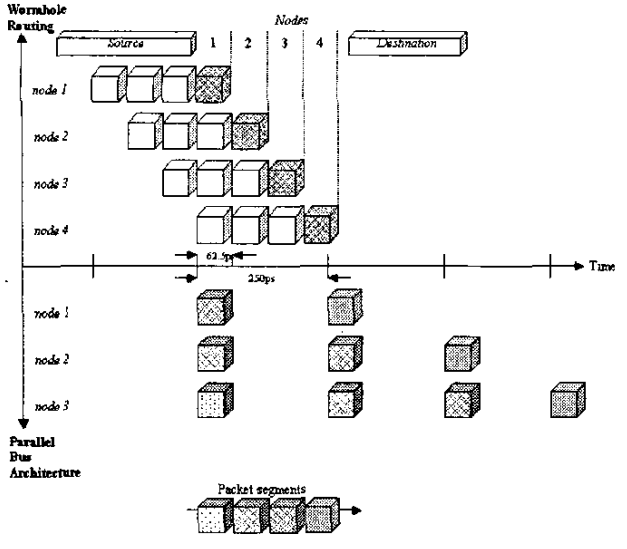


Fig. 6. Message timing (3D bus vs. parallel bus)

Throughput is defined as the rate in which the packets are exiting the bus for a certain message size per second. The resulting throughput ($Tp$) equation is

$$Tp_{3D} = \frac{M}{[T * ((\frac{M}{w} - 1) + D) + D * T(n)]} \quad (2)$$

We compare the performance of the 3D bus with that of the four parallel buses. This is mainly because 3D bus resembles to four parallel buses with some extra exits/links in every corner. The latency of this parallel bus can be given as

$$L_{4b} = \frac{M}{w} * T * c * \frac{1}{4} \quad (3)$$

which describes the propagation delay for the number of packets in a message divided by 4 because we are using 4 parallel buses. Here it is assumed that each cube link is 1 cm long and the four parallel buses have the same length as the 3D bus formed by $c$ cubes.

The throughput for the four parallel buses is obtained by dividing the message size by the latency

$$Tp_{4b} = \frac{M}{\frac{M}{w} * T * c * \frac{1}{4}} = \frac{4 * w}{T * D} \quad (4)$$

### D. Results

We use the equations developed above to calculate the latency and throughput measures. We are also interested in determining the optimal number of cubes that are needed to form the 3D bus and also the channel widths. Figure 7 shows the latency ratio (parallel bus to 3D bus) against the number of cubes for several channel widths when message size is kept at 1024 B. The figure implies that 3D bus with 5 cubes or more has lower latency than the four parallel buses. It also reveals that the ratio improves with lower channel width.

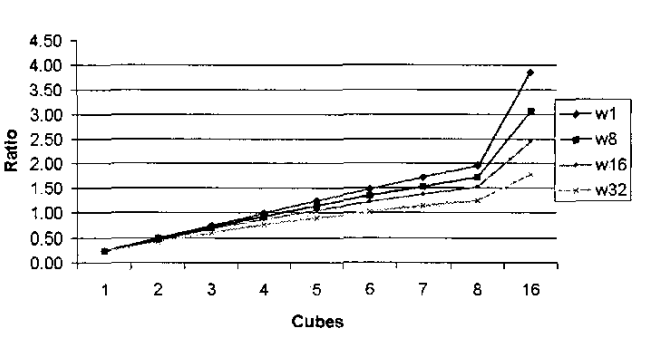


Fig. 7. Latency Ratio (Message size 1024B)

The latency increases with message size as shown in Fig. 8 when bus length is 8 cubes. For messages larger than 256 B, 3D bus has lower latency compared to 4 parallel buses with the same channel width.

Figure 9 shows the throughput ratio (3D bus to parallel bus) against the number of cubes for several channel widths when message size is kept at 1024 B. The ratio improves with increasing bus length this is due to the pipeline-like mechanism used by the 3D bus, and also store and forward type architecture used by the parallel bus. In addition, as the

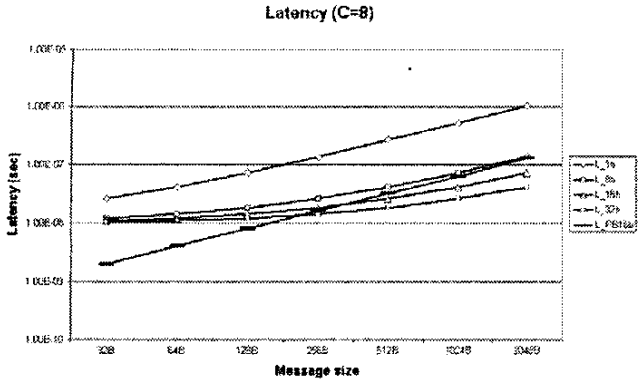


Fig. 8. Latency vs message size

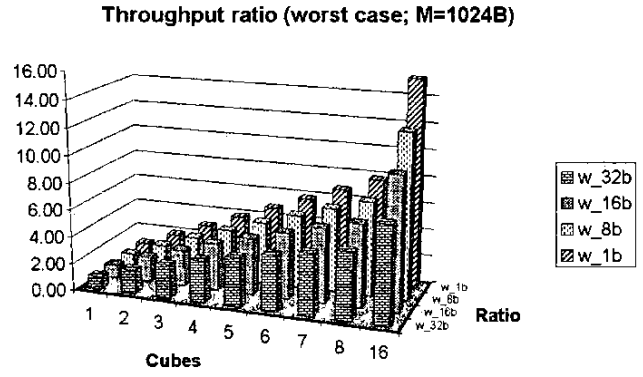


Fig. 9. Throughput Ratio (Message size 1024B)

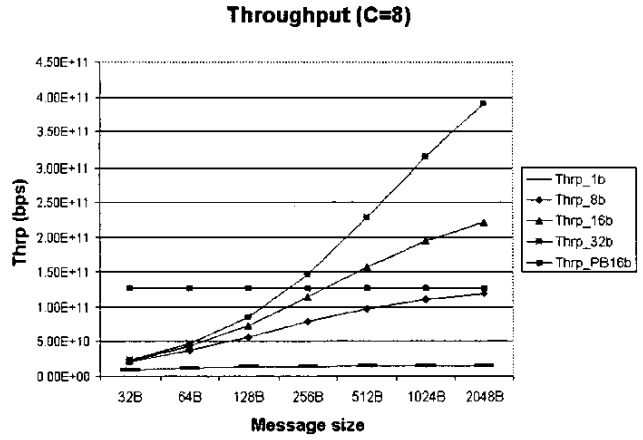


Fig. 10. Throughput vs message size

channel width increases there is some reduction in throughput ratio however in all scenarios, the 3D bus is still proven to reach better performance. The throughput of the 3D bus improves with increasing message size as shown in Fig. 10. This is in result of filling the packets on the links of the 3D bus. In contrast, the throughput stays the same for the parallel bus due to store and forward type architecture. Our 3D bus interconnect reached almost 400Gbps with 32 bit channel size and 2048B message size. Currently there are no memories which operate in throughput as high as 400Gbps. Significantly, our approach prop to use existing high bandwidth memories (such as Rambus) [6] while we design the memory interface to efficiently employ their internal banks. By taking this cost-effective approach we assure that the interconnection bus will be effortlessly integrated into existing network linecard boards and other on-board processor-memory architectures.

| CSIX | HyperTransport | PCI Express | SPI 4.2 | 3D-bus |
|---|---|---|---|---|
| 3.2 | 102.4 | 4.26 | 12.8 | **400** |

Fig. 11. Throughput comparison with current interconnect technologies

## IV. Conclusions and Future Work

An interconnection architecture for network line cards is presented. The 3D bus architecture allows the multiple processing elements on the line card to access multiple memory modules. Besides the line card, the 3D bus structure can also be used within network processors as an on-chip communication mechanism between processing elements, memory and peripherals. This can improve the performance of the current implementations such as Intel IXP2800, where 8 micro engines compete for one bus. The results show that the throughput significantly improved when compared with parallel bus or other contestants structures currently in the market. Future work include memory and PE interfaces which are able to supply the resulted bandwidth to memory and processing modules.

## References

[1] D. Halliday, "The evolution of mezzanine modules for next-generation telecom architectures", *CompactPCI-Systems*, June 2003.
[2] W. J. Dally, "Performance analysis of k-ary n-cube interconnection networks", *IEEE Tran. on Computers*, vol. 39, no. 6, pp. 775-785, 1990.
[3] K. Marquardt, "Hitting the 10-Gbit Mark with SPI-4.2", (web:www.commsdesign.com/design_corner/OEG20020910S0010).
[4] CSIX Interface, White Paper (web:www.altera.com /products/ip /communications/csix/ipm-index.jsp).
[5] HyperTransport Consortium, "HyperTransport technology: Simplifying system design", Oct. 2002 (web: http://www.hypertransport.org).
[6] "Rambus DRAM for OC192 Data Rate Line Card Applications", Rambus Inc., 2000.
[7] "IXP2800 Intel Network Processor IP Forwarding Benchmark Full Disclosure Report for OC192-POS", White Paper, Intel corp., Oct. 30, 2003.
[8] PCI Special Interest Group, "PCI local bus specification, revision 2.2", Dec. 1998.